Low-Code-Entwicklung

AUER FLORIAN

HERZOG MAX

LOIDL PATRICK

POLZER LEO

Agenda

1. Was ist Low-Code?

- Generelle Übersicht
- Vergleich zu herkömmlichen Entwicklungsmethoden
- Vor- und Nachteile

2. Low-Code am Beispiel von Mendix

- Entwicklungsumgebung <-> WebApp
- Domain Modell
- Microflows

3. Einsatz in der Wirtschaft

- Genereller Überblick
- Vergleich Low-Code vs Code
- Unternehmen die Low-Code nutzen

4. Einsatz in Bildung und Alltag

- Low-Code für Lehrer und Schüler
- Citizen Developer
- Zukunftsausblick

Was ist Low-Code

- Methode zur Softwareentwicklung
- Anwendungen mit minimalem Code
- Visuelle Entwicklungsumgebung
- Bietet vorgefertigte Bausteine
- Entwicklungsprozesse beschleunigen

Low-Code vs Traditionelle Entwicklungsmethoden

LOW CODE

- Schnelle App-Entwicklung
- Grafische Benutzeroberfläche
- Reduzierte Komplexität

TRADITIONELLE ENTWICKLUNGSMETHODEN

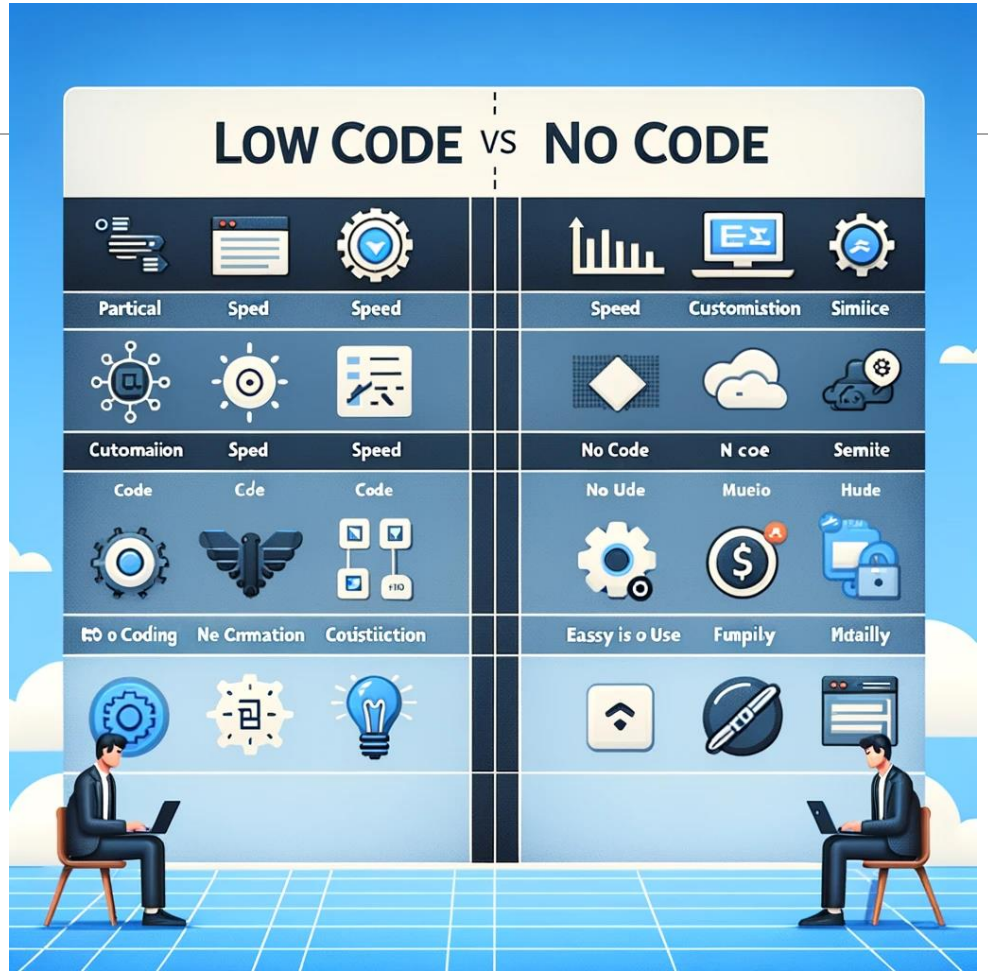
- Kenntnisse in Programmiersprachen
- Manuellen Code schreiben
- Flexible Softwaregestaltung
- Längere Entwicklungszeiten

Merkmale

- Visuelle Entwicklungsumgebung
- Einfache Integration und Anpassung
- Schnelle Entwicklung und Bereitstellung
- Breites Spektrum an Nutzern

Low-Code vs No-Code

- Minimaler Code
- Schnelligkeit
- Anpassungsfähigkeit



- Kein Code
- Einfachheit
- Vollständig visuelle Schnittstelle
- Benutzerfreundlichkeit

Vorteile und Nachteile von Low-Code

VORTEILE

- Schnelle Entwicklung
- Benutzerfreundlichkeit
- Kosteneffizienz
- Wartung

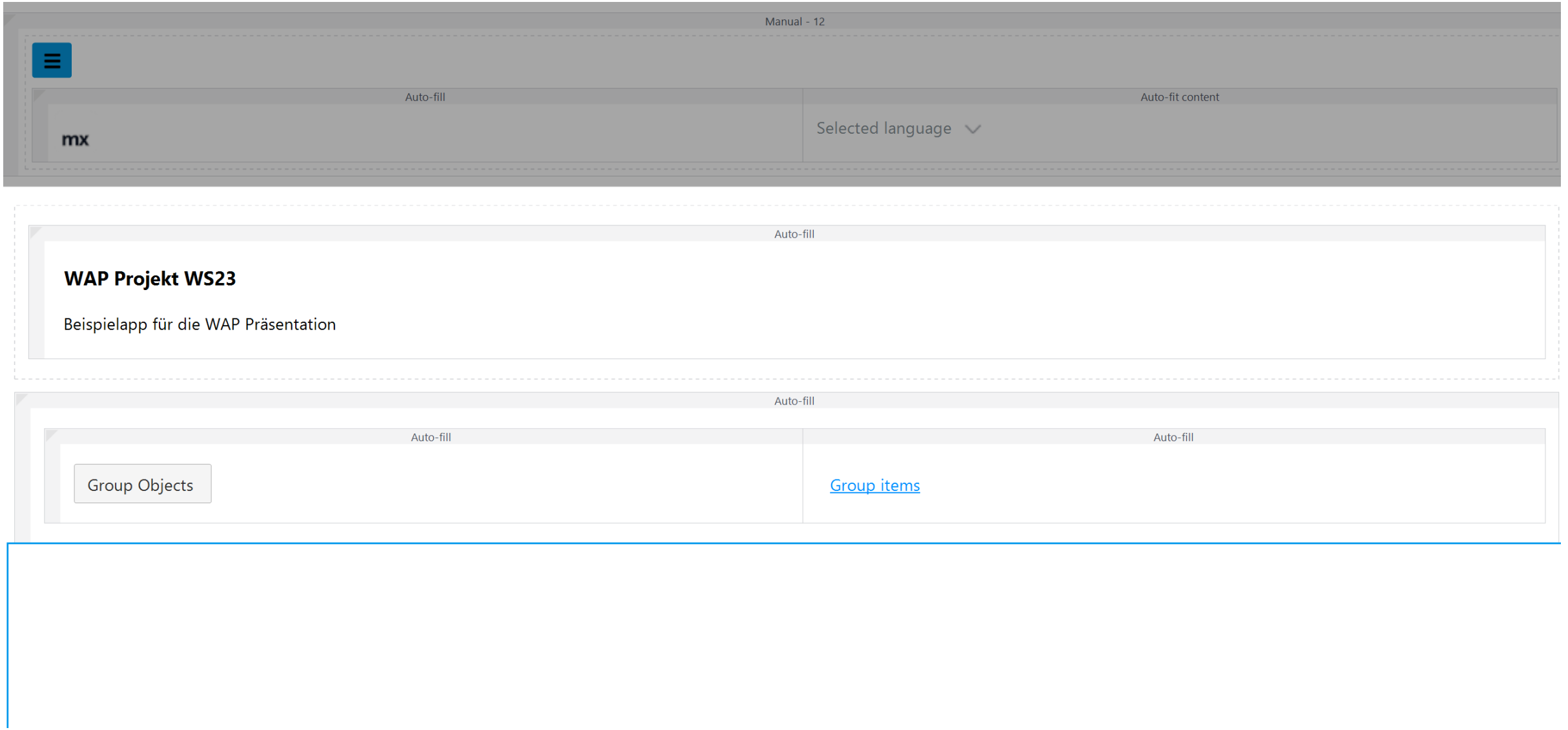
NACHTEILE

- Begrenzung Anpassungsfähigkeit
- Leistungsbeschränkungen
- Abhängigkeit von Plattformanbietern
- Sicherheitsbedenken

Mendix

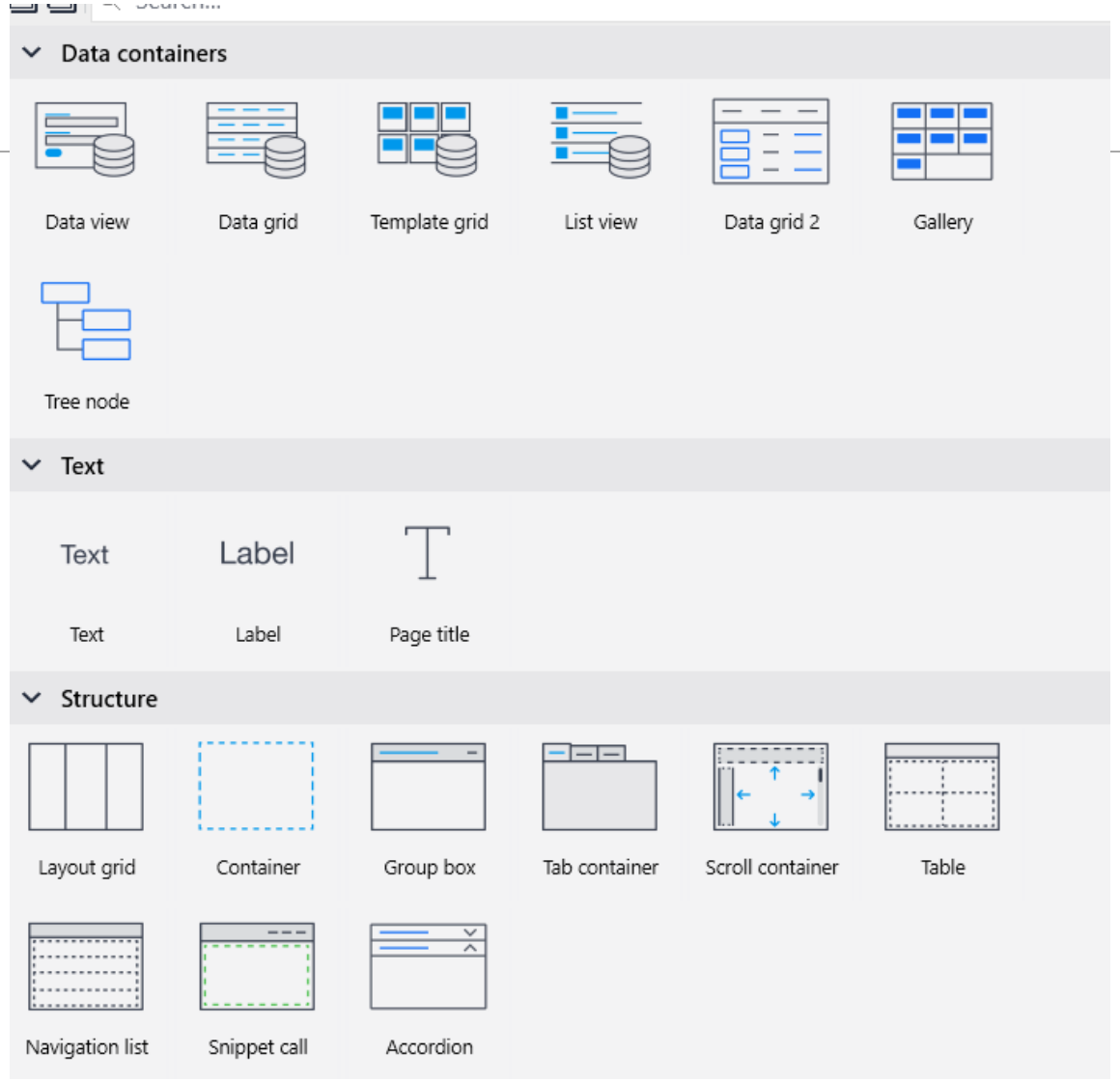
- Marktführende Low-Code Plattform
- 2005 in den Niederlanden gegründet
- 2018 von Siemens für 730 Millionen € gekauft
- Mendix aktuell:
  - 300 000 aktive Entwickler
  - 4 000 Kunden
  - 50 Millionen Endnutzer
  - 200 000 entwickelte Apps

Mendix Entwicklungsumgebung

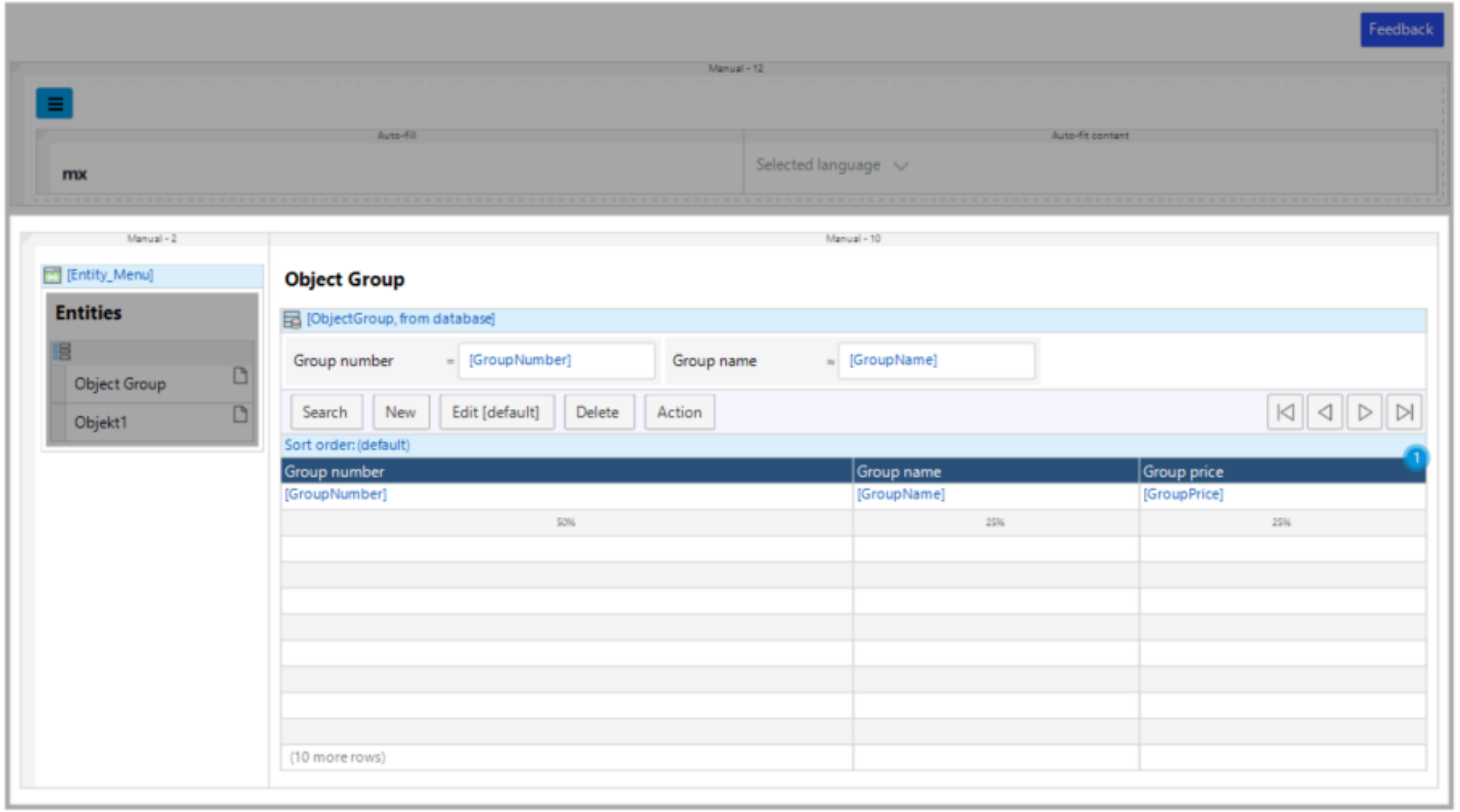


Browser Live-Umgebung

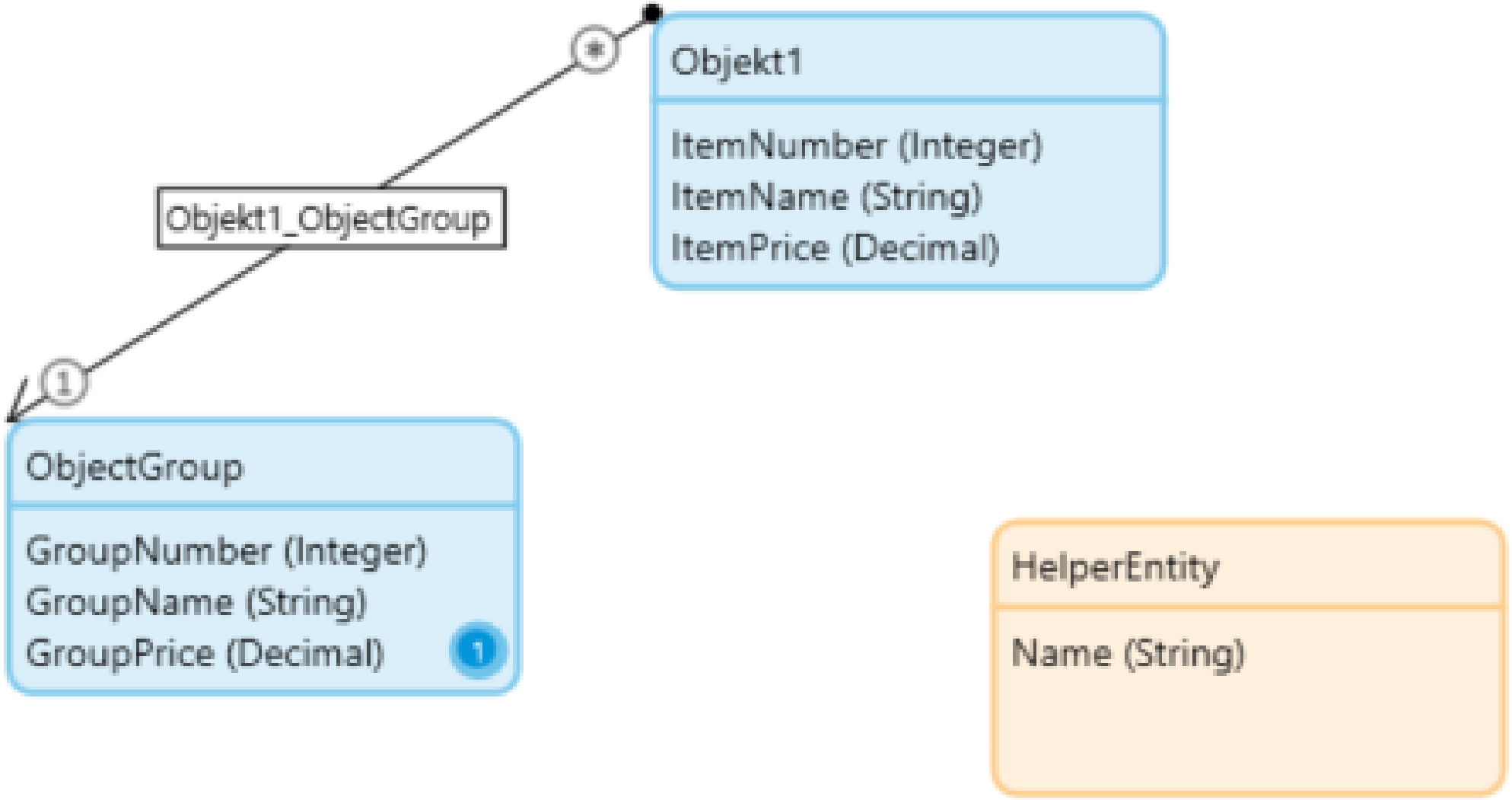
Widgets


Data containers
Data view
Data grid
Template grid
List view
Data grid 2
Gallery
Tree node
Text
Text
Label
Page title
Structure
Layout grid
Container
Group box
Tab container
Scroll container
Table
Navigation list
Snippet call
Accordion

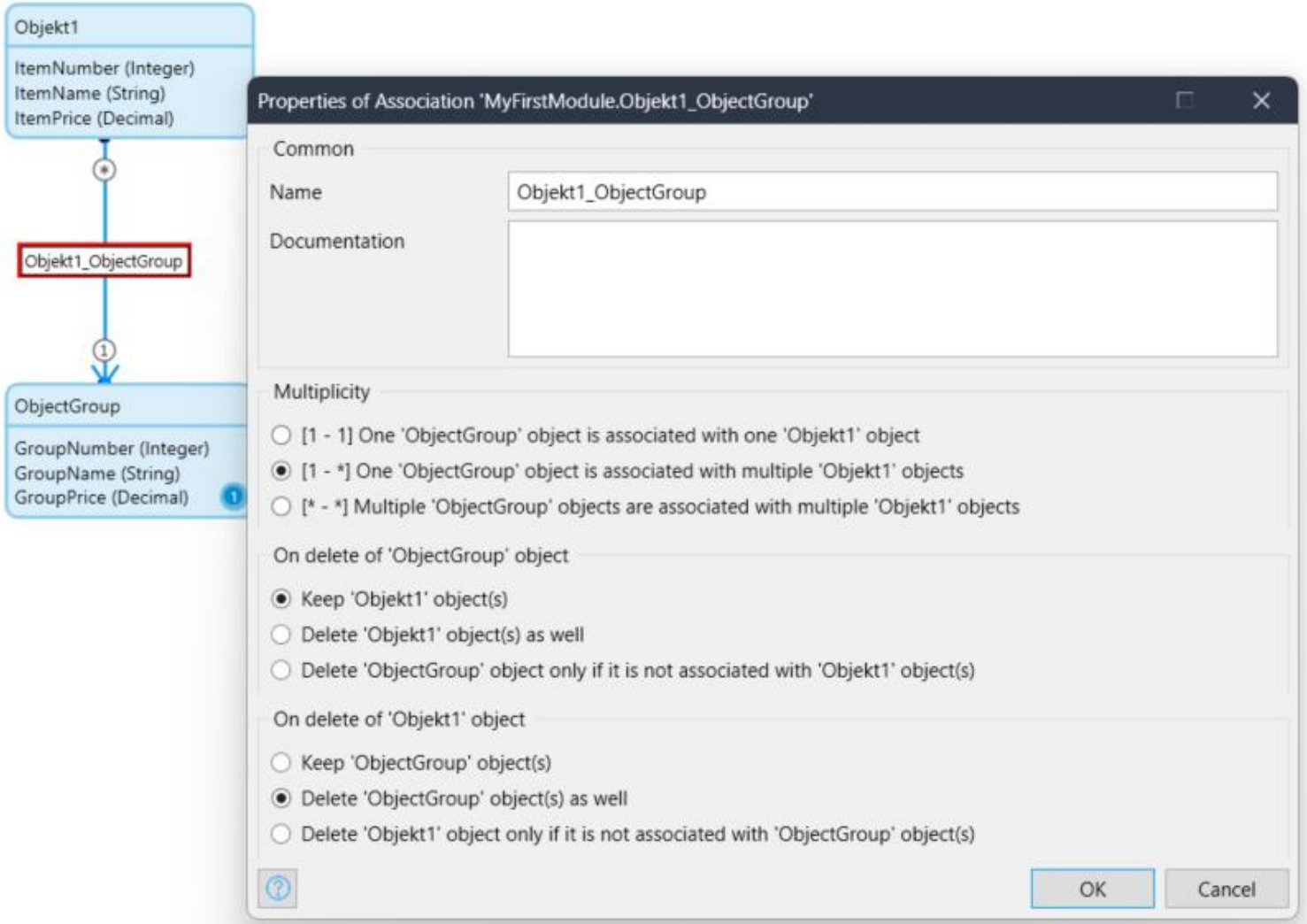
Grafische Benutzeroberfläche



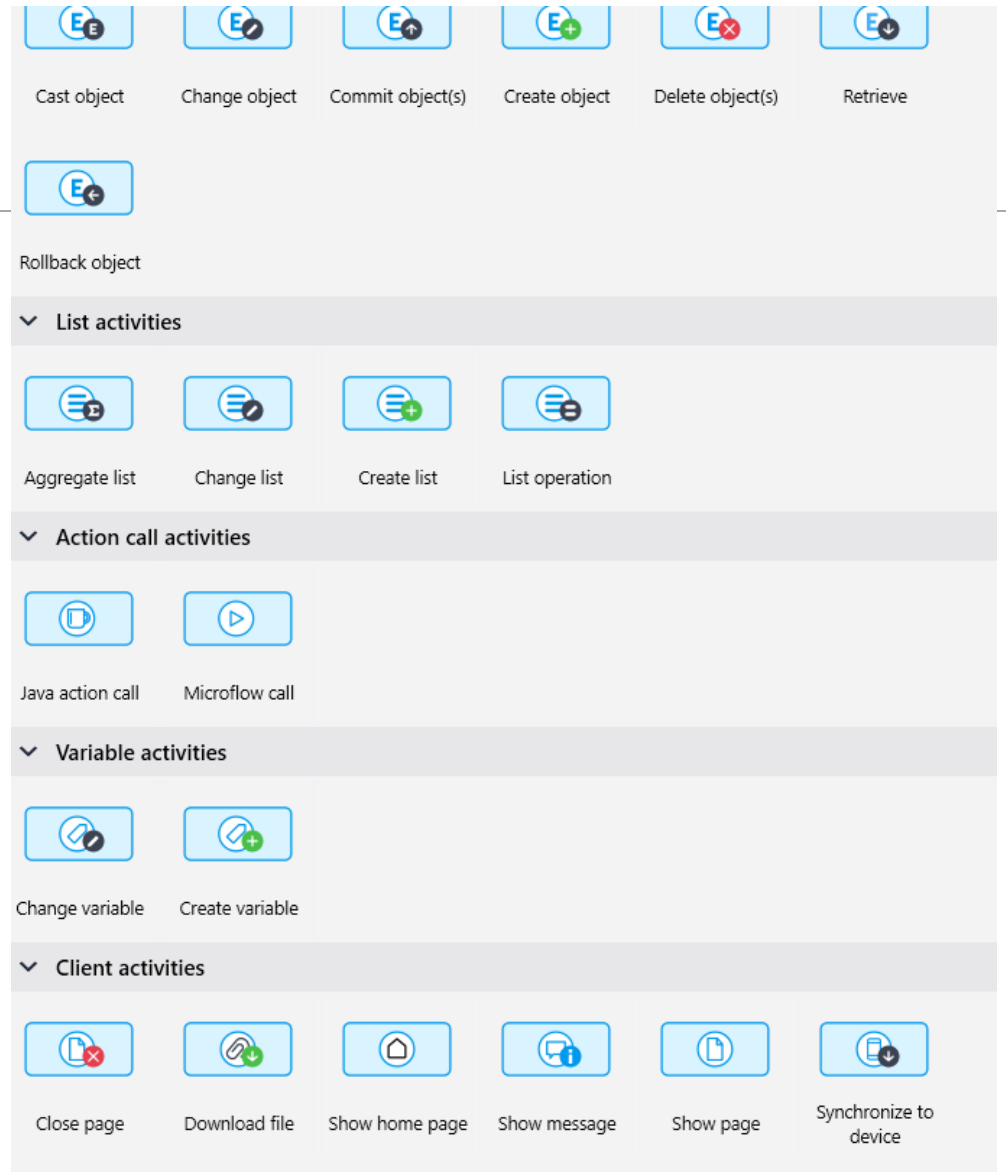
Domain Modell



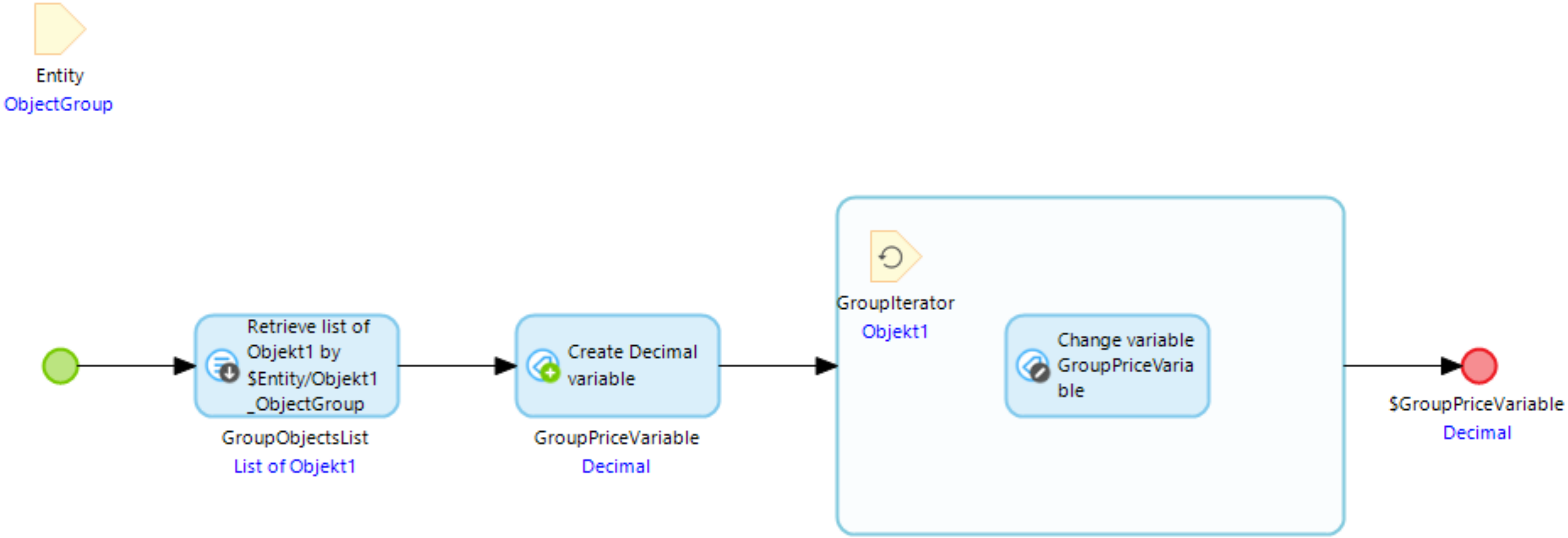
Assoziationen



Microflows


Cast object
Change object
Commit object(s)
Create object
Delete object(s)
Retrieve
Rollback object
List activities
Aggregate list
Change list
Create list
List operation
Action call activities
Java action call
Microflow call
Variable activities
Change variable
Create variable
Client activities
Close page
Download file
Show home page
Show message
Show page
Synchronize to device

Beispiel eines Microflows



Einsatz von Low-Code in der Wirtschaft

- Beschleunigte Anwendungsentwicklung
- Effiziente Ressourcennutzung
- Integration von Legacy Systemen
- Microservices-Architektur
- Vergleich Low-Code vs Nicht Low-Code
- Unternehmen die Low-Code nutzen

Beschleunigte Anwendungsentwicklung

- Schnell auf sich ändernde Marktanforderungen reagieren
- Agile Produktentwicklung
- Drag and Drop Development
- Spart Zeit und Kosten

Effiziente Ressourcennutzung

- Schnelles Coding für Mitarbeiter ohne Erfahrung
- Beispiel Santander Bank
- Automatisierung von Routineaufgaben
- Mehr Zeit für strategische Aufgaben

Integration von Legacy-Systemen

- Hohe Kosten für Aufrechterhaltung
- Schrittweise Modernisierung und Transformation
- Vorteil für komplexe IT-Infrastrukturen

Microservices-Architektur

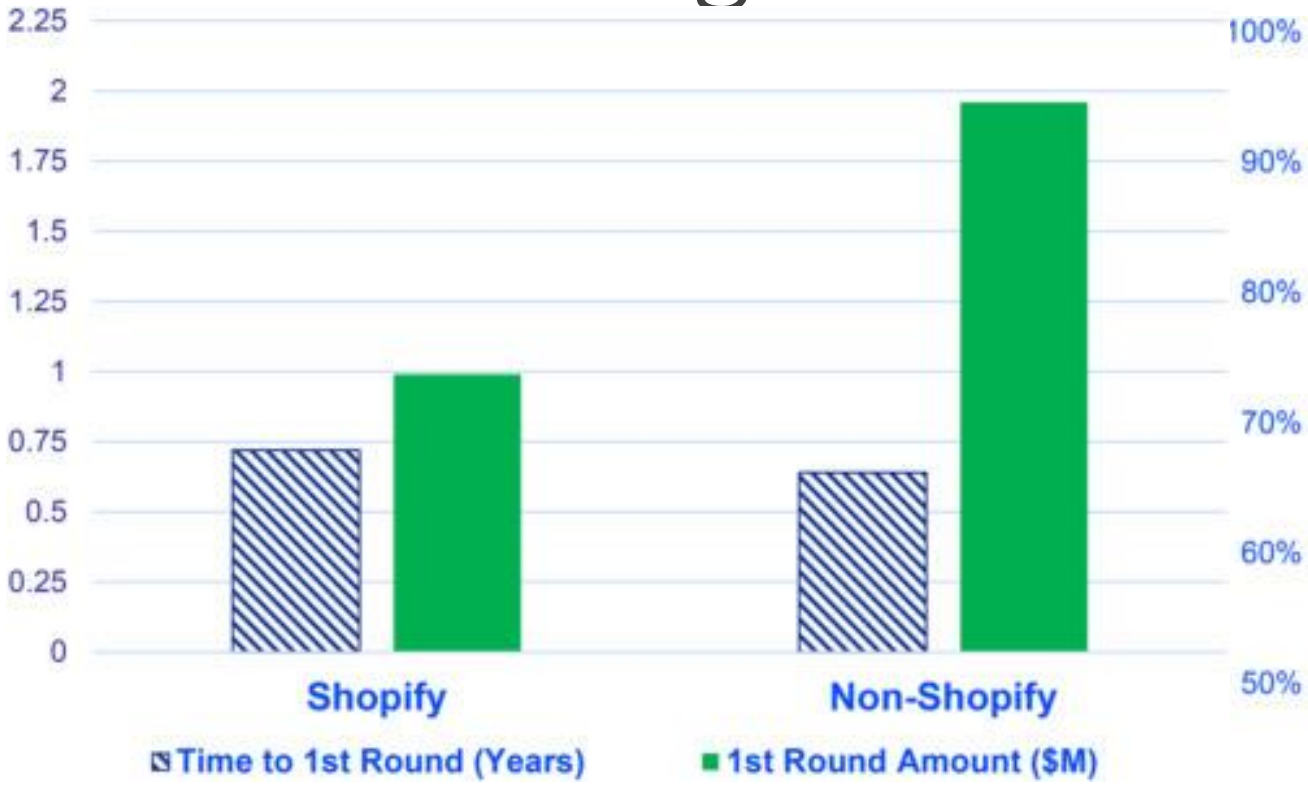
Generell:

- Leicht wartbare Strukturen
- Wiederverwendbarkeit
- Verbesserte Kundenerfahrung

Use Case

- Proof of concept
- API Management
- Geringere Ausgaben

Vergleich Low-Code vs Code



Amount and time to first financing round, by Shopify and non-Shopify startups

Startup final status, by Shopify and non-Shopify startups

Gary Dushnitsky, Bryan K. Stroube
Quelle: https://www.sciencedirect.com/science/article/pii/S2352673421000299

Unternehmen die Low-Code nutzen

- Colliers International Group
- United Health Group
- Royal BAM Group
- Siemens
- Lufthansa
- Santander

Low-Code in Bildung und Alltag

- Low-Code für Lehrende
- Low-Code für Schüler
- Citizen Developer

Low-Code für Lehrende

- Entwicklung von eigenen kleinen Lernprogrammen
  - Lernen bei Schülern kann individueller werden
- Vereinfachung/Automatisierung administrativer Aufgaben
- Entlastung bei Routineaufgaben
- Kann für Gruppenarbeiten eingesetzt werden

Low-Code für Schüler

- Leichterer Übergang von z.B. Scratch oder Lego Roboter
- Leichterer Umgang als mit traditionellen Programmiersprachen (Visuelle Komponente)
  - Reduzierung kognitiver Belastung
  - Bessere Wissensaufnahme

Citizen Developer

- Leute, die Programmieren, ohne das (professionell) gelernt zu haben
- Durch Low-Code/No-Code immer verbreiteter
- Genaue Auswirkungen noch nicht bekannt:
  - Kennen sich wenig/nicht mit Dingen wie Laufzeitkomplexität oder Sicherheit aus

Zukunftsausblick

- Zukunft für Nicht-Programmierer
- Zukunft für die Wirtschaft

Zukunft für Nicht-Programmierer

- Großes Wachstum von Citizen Development
- Demokratisierung von Programmieren
  - Nicht-Programmierer können leichter Programme verstehen, ändern und selbst erstellen

Zukunft für die Wirtschaft

- Low-Code wird immer verbreiteter
  - Bis 2026 sollen 80% der Anwendungen mit Low-Code sein [1]
- Wirkt gegen Fachkräftemangel
  - Citizen Developer statt prof. Programmierern
- Noch zu neu

Quelle: [1] Gartner Report: https://www.gartner.com/en/newsroom/press-releases/2022-12-13-gartner-forecasts-worldwide-low-code-development-technologies-market-to-grow-20-percent-in-2023